Online Bayesian Kernel Regression from Nonlinear Mapping of Observations

Matthieu Geist, Olivier Pietquin, Gabriel Fricout

ONLINE BAYESIAN KERNEL REGRESSION FROM NONLINEAR MAPPING OF OBSERVATIONS

Matthieu Geist[1,2]*, Olivier Pietquin*[1] *and Gabriel Fricout*[2]

[1]SUPELEC, IMS Research Group, Metz, France
[2]ArcelorMittal Research, MCE Department, Maizières-lès-Metz, France

ABSTRACT

In a large number of applications, engineers have to estimate a function linked to the state of a dynamic system. To do so, a sequence of samples drawn from this unknown function is observed while the system is transiting from state to state and the problem is to generalize these observations to unvisited states. Several solutions can be envisioned among which regressing a family of parameterized functions so as to make it fit at best to the observed samples. However classical methods cannot handle the case where actual samples are not directly observable but only a nonlinear mapping of them is available, which happen when a special sensor has to be used or when solving the Bellman equation in order to control the system. This paper introduces a method based on Bayesian filtering and kernel machines designed to solve the tricky problem at sight. First experimental results are promising.

1. INTRODUCTION

In a large number of applications, engineers have to estimate a function linked to the state of a dynamic system. This function is generally qualifying the system (*e.g.* the magnitude of the eletro-magnetic field in a wifi network according to the power emitted by each antenna in the network). To do so, a sequence of samples drawn from this unknown function is observed while the system is transiting from state to state and the problem is to generalize these observations to unvisited states. Several solutions can be envisioned among which regressing a family of parameterized functions so as to make it fit at best to the observed samples. This is a well known problem that is addressed by many standard methods in the literature such as the kernel machines methods [1] or neural networks [2].

Yet, several problems are not usually handled by standard techniques. For instance, actual samples are sometimes not directly observable but only a nonlinear mapping of them is available. This is the case when a special sensor has to be used (*e.g.* measuring a temperature using a spectrometer or a thermocouple). This is also the case when solving the Bellman equation in a Markovian decision process with unknown deterministic transitions [3]. This is important for (asynchronous and online) dynamic programming, and more generally for control theory. Another problem is to handle online regression, that is incrementally improve the regression results as new samples are observed by recursively updating previously computed parameters.

In this paper we propose a method, based on the Bayesian filtering paradigm and kernel machines, for online regression of non-linear mapping of observations. We have chosen here to describe a quite general formulation of the problem in order to appeal a broader audience. Indeed the technique introduced below handles well non-linearities in a derivative free way and it should be useful in other fields.

1.1. Problem statement

Through the rest of this paper, $\mathbf{x}$ will denote a column vector and x a scalar. Let $\mathbf{x} = (\mathbf{x}^1, \mathbf{x}^2) \in \mathcal{X} = \mathcal{X}^1 \times \mathcal{X}^2$ where $\mathcal{X}^1$ (resp. $\mathcal{X}^2$) is a compact set of $\mathbb{R}^n$ (resp. $\mathbb{R}^m$). Let $t : \mathcal{X}^1 \times \mathcal{X}^2 \to \mathcal{X}^1$ be a nonlinear transformation (transitions in case of dynamic systems) which will be observed. Let g be a nonlinear mapping such that $g : f \in \mathbb{R}^{\mathcal{X}} \to g_f \in \mathbb{R}^{\mathcal{X} \times \mathcal{X}^1}$. The aim here is to approximate sequentially the nonlinear function $f : \mathbf{x} \in \mathcal{X} \to f(\mathbf{x}) = f(\mathbf{x}^1, \mathbf{x}^2) \in \mathbb{R}$ from samples

$$\left(\mathbf{x}_k, \mathbf{t}_k = t(\mathbf{x}_k^1, \mathbf{x}_k^2), y_k = g_f(\mathbf{x}_k^1, \mathbf{x}_k^2, \mathbf{t}_k)\right)_k$$

by a function $\hat{f}_\theta(\mathbf{x}) = \hat{f}_\theta(\mathbf{x}^1, \mathbf{x}^2)$ parameterized by the vector θ. Here the output is scalar, however the proposed method can be straightforwardly extended to vectorial outputs. Note that as

$$\left(\mathbb{R}^{\mathcal{X}}\right)^{\mathbb{R}^{\mathcal{X}}} \subset \left\{ g \in \left(\mathbb{R}^{\mathcal{X} \times \mathcal{X}^1}\right)^{\mathbb{R}^{\mathcal{X}}} | g_f : \mathbf{x} \in \mathcal{X} \to g_f(\mathbf{x}, t(\mathbf{x})) \right\}$$

this problem statement is quite general. Thus the work presented in this paper can be considered with $g : f \in \mathbb{R}^{\mathcal{X}} \to g_f \in \mathbb{R}^{\mathcal{X}}$ (g being known analytically), this case being more specific. The interest of this particular formulation is that a part of the nonlinearities has to be known analytically (the mapping g) whereas the other part can be just observed (the t function).

1.2. Outline

A kernel-based regression is used, namely the approximation is of the form $\hat{f}_\theta(\mathbf{x}) = \sum_{i=1}^{p} \alpha_i K(\mathbf{x}, \mathbf{x}_i)$ where K is a kernel, that is a continuous, symmetric and positive semi-definite function. The parameter vector θ contains the weights $(\alpha_i)_i$, and possibly the centers $(\mathbf{x}_i)_i$ and some parameters of the kernel (*e.g.* the variance for Gaussian kernels). These methods rely on the Mercer theorem [4] which states that each kernel is a dot product in a higher dimensional space. More precisely, for each kernel K, it exists a mapping $\varphi : \mathcal{X} \to \mathcal{F}$ ($\mathcal{F}$ being called the feature space) such that $\forall \mathbf{x}, \mathbf{y} \in \mathcal{X}$, $K(\mathbf{x}, \mathbf{y}) = \langle \varphi(\mathbf{x}), \varphi(\mathbf{y}) \rangle$. Thus, any linear regression algorithm which only uses dot products can be cast by this *kernel trick* into a nonlinear one by implicitly mapping the original space $\mathcal{X}$ to a higher dimensional one. Many approaches to kernel regression can be found in the literature, the most classical being the Support Vector Machines (SVM) framework [4]. There are quite fewer Bayesian approaches, nonetheless the reader can refer to [5] or [6] for interesting examples. To our knowledge, none of them is designed to handle the regression problem described in this paper.

As a preprocessing step a prior on the kernel to be used is put (*e.g.* a Gaussian kernel with a specific width), and a dictionary method [7] is used to compute an approximate basis of $\varphi(\mathcal{X})$. This gives a fair sparse and noise-independent initialisation for the number of kernels to be used and their centers. Following [8], the regression problem is cast into a state-space representation

$$\begin{aligned} \theta_{k+1} &= \theta_k + \mathbf{v}_k \\ y_k &= g_{\hat{f}_{\theta_k}}(\mathbf{x}_k^1, \mathbf{x}_k^2, \mathbf{t}_k) + n_k \end{aligned} \tag{1}$$

where $\mathbf{v}_k$ is an artificial process noise and n_k is an observation noise. A Sigma Point Kalman Filter (SPKF) [9] is used to sequentially estimate the parameters, as the samples $(\mathbf{x}_k, t(\mathbf{x}_k), y_k)_k$ are observed.

In the following sections, the SPKF approach and the dictionary method will be first briefly presented. The proposed algorithm, which is based on the two forementioned methods, will then be exhibited, and the first experimental results will be shown. Eventually, future works will be sketched.

2. BACKGROUND

2.1. Bayesian Filtering Paradigm

The problem of Bayesian filtering can be expressed in its state-space formulation:

$$\begin{aligned} \mathbf{s}_{k+1} &= \psi_k(\mathbf{s}_k, \mathbf{v}_k) \\ y_k &= h_k(\mathbf{s}_k, n_k) \end{aligned} \tag{2}$$

The objective is to sequentially infer $E[\mathbf{s}_k | y_{1:k}]$, the expectation of the hidden state $\mathbf{s}_k$ given the sequence of observations up to time k, $y_{1:k}$, knowing that the state evolution is driven by the possibly nonlinear mapping ψ_k and the process noise $\mathbf{v}_k$. The observation y_k, through the possibly nonlinear mapping h_k, is a function of the state $\mathbf{s}_k$, corrupted by an observation noise n_k. If the mappings are linear and if the noises $\mathbf{v}_k$ and n_k are additional Gaussian noises, the optimal solution is given by the Kalman filter [10]: quantities of interest are random variables, and inference (that is prediction of these quantities and correction of them given a new observation) is done online by propagating sufficient statistics through linear transformations. If one of these two hypothesis does not hold, approximate solutions exist. See [11] for a survey on Bayesian filtering.

The Sigma Point framework [9] is a nonlinear extension of Kalman filtering (random noises are still Gaussian). The basic idea is that it is easier to approximate a probability distribution than an arbitrary nonlinear function. Recall that in the Kalman filter framework, basically linear transformations are applied to sufficient statistics. In the Extended Kalman Filter (EKF) approach, nonlinear mappings are linearized. In the Sigma Point approach, a set of so-called sigma points are deterministically computed using the estimates of mean and of covariance of the random variable of interest. These points are representative of the current distribution. Then, instead of computing a linear (or linearized) mapping of the distribution of interest, one calculates a nonlinear mapping of these sigma points, and use them to compute sufficient statistics of interest for prediction and correction equations, that is to approximate the following distributions:

$$\begin{aligned} p(\mathbf{S}_k | Y_{1:k-1}) &= \int_{\mathcal{S}} p(\mathbf{S}_k | \mathbf{S}_{k-1}) p(\mathbf{S}_{k-1} | Y_{1:k-1}) d\mathbf{S}_{k-1} \\ p(\mathbf{S}_k | Y_{1:k}) &= \frac{p(Y_k | \mathbf{S}_k) p(\mathbf{S}_k | Y_{1:k-1})}{\int_{\mathcal{S}} p(Y_k | \mathbf{S}_k) p(\mathbf{S}_k | Y_{1:k-1}) d\mathbf{S}_k} \end{aligned}$$

Note that SPKF and classical Kalman equations are very similar. The major change is how to compute sufficient statistics (directly for Kalman, through sigma points computation for SPKF). Table 1 sketches a SPKF update in the case of additive noise, based on the state-space formulation, and using the standard Kalman notations: $\mathbf{s}_{k|k-1}$ denotes a prediction, $\mathbf{s}_{k|k}$ an estimate (or correction), $P_{\mathbf{s}_k, y_k}$ a covariance matrix, $\bar{n}_k$ a mean and k is the discrete time index. The reader can refer to [9] for details.

2.2. Dictionary method

As said in section 1, the kernel trick corresponds to a dot product in a higher dimensional space $\mathcal{F}$, associated with a mapping φ. By observing that although $\mathcal{F}$ is a (very) higher dimensional space, $\varphi(\mathcal{X})$ can be a quite smaller em-

Table 1. SPKF Update

inputs: $\bar{\mathbf{s}}_{k-1|k-1}$, $P_{\mathbf{s}_{k-1|k-1}}$
outputs: $\bar{\mathbf{s}}_{k|k}$, $P_{\mathbf{s}_{k|k}}$

Sigma points computation:
Compute deterministically sigma-point set $\mathbf{S}_{k-1|k-1}$ from $\bar{\mathbf{s}}_{k-1|k-1}$ and $P_{\mathbf{s}_{k-1|k-1}}$

Prediction step:
Compute $\mathbf{S}_{k|k-1}$ from $\psi_k(\mathbf{S}_{k-1|k-1}, \bar{\mathbf{v}}_k)$ and process noise covariance
Compute $\bar{\mathbf{s}}_{k|k-1}$ and $P_{\mathbf{s}_{k|k-1}}$ from $\mathbf{S}_{k|k-1}$

Correction step:
Observe y_k
$Y_{k|k-1} = h_k(\mathbf{S}_{k|k-1}, \bar{n}_k)$
Compute $\bar{y}_{k|k-1}$, $P_{y_{k|k-1}}$ and $P_{\mathbf{s}_{k|k-1}, y_{k|k-1}}$ from $\mathbf{S}_{k|k-1}$, $Y_{k|k-1}$ and observation noise covariance
$K_k = P_{\mathbf{s}_{k|k-1}, y_{k|k-1}} P^{-1}_{y_{k|k-1}}$
$\bar{\mathbf{s}}_{k|k} = \bar{\mathbf{s}}_{k|k-1} + K_k(y_k - \bar{y}_{k|k-1})$
$P_{\mathbf{s}_{k|k}} = P_{\mathbf{s}_{k|k-1}} - K_k P_{y_{k|k-1}} K_k^T$

bedding, the objective is to find a set of p points in $\mathcal{X}$ such that $\varphi(\mathcal{X}) \simeq \text{Span}\{\varphi(\tilde{\mathbf{x}}_1), \ldots, \varphi(\tilde{\mathbf{x}}_p)\}$.

This procedure is iterative. Suppose that samples $\mathbf{x}_1, \mathbf{x}_2, \ldots$ are sequentially observed. At time k, a dictionary $\mathcal{D}_{k-1} = (\tilde{\mathbf{x}}_j)_{j=1}^{m_{k-1}} \subset (\mathbf{x}_j)_{j=1}^{k-1}$ of m_{k-1} elements is available where by construction feature vectors $\varphi(\tilde{\mathbf{x}}_j)$ are approximately linearly independent in $\mathcal{F}$. A sample $\mathbf{x}_k$ is then observed, and is added to the dictionary if $\varphi(\mathbf{x}_k)$ is linearly independent on $\mathcal{D}_{k-1}$. To test this, weights $\mathbf{a} = (a_1, \ldots, a_{m_{k-1}})^T$ have to be computed so as to verify

$$\delta_k = \min_{\mathbf{a} \in \mathbb{R}^{m_{k-1}}} \left\| \sum_{j=1}^{m_{k-1}} a_j \varphi(\tilde{\mathbf{x}}_j) - \varphi(\mathbf{x}_k) \right\|^2 \qquad (3)$$

Formally, if $\delta_k = 0$ then the feature vectors are linearly dependent, otherwise not. Practically an approximate dependence is allowed, and δ_k will be compared to a predefined threshold ν determining the quality of the approximation (and consequently the sparsity of the dictionary). Thus the feature vectors will be considered as approximately linearly dependent if $\delta_k \leq \nu$.

By using the kernel trick and the bilinearity of dot products, equation (3) can be rewritten as

$$\delta_k = \min_{\mathbf{a}} \left\{ \mathbf{a}^T \tilde{K}_{k-1} \mathbf{a} - 2\mathbf{a}^T \tilde{\mathbf{k}}_{k-1}(\mathbf{x}_t) + K(\mathbf{x}_k, \mathbf{x}_k) \right\} \qquad (4)$$

where $(\tilde{K}_{k-1})_{i,j} = K(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j)$ is a $m_{k-1} \times m_{k-1}$ matrix and $(\tilde{\mathbf{k}}_{k-1}(\mathbf{x}))_i = K(\mathbf{x}, \tilde{\mathbf{x}}_i)$ is a $m_{k-1} \times 1$ vector. If $\delta_k > \nu$, $\mathbf{x}_k = \tilde{\mathbf{x}}_{m_k}$ is added to the dictionary, otherwise not. Equation (4) admits the analytical solution $\mathbf{a}_k = \tilde{K}_{k-1}^{-1} \tilde{\mathbf{k}}_{k-1}(\mathbf{x}_k)$. Moreover it exists a computationally efficient algorithm which uses the partitioned matrix inversion formula to construct this dictionary. See [7] for details.

Thus, by choosing a prior on the kernel to be used, and by applying this algorithm to a set of N points randomly sampled from $\mathcal{X}$, a sparse set of good candidates to the kernel regression problem is obtained. This method is theoretically well founded, easy to implement, computationally efficient and it does not depend on kernels nor space's topology. Note that, despite the fact that this algorithm is naturally online, this dictionary cannot be built (straightforwardly) while estimating the parameters, since the parameters of the chosen kernels (such as mean and deviation for Gaussian kernels) will be parameterized as well. Observe that only bounds on $\mathcal{X}$ have to be known in order to compute this dictionary, and not the samples used for regression.

3. PROPOSED ALGORITHM

Recall that the objective here is to sequentially approximate a nonlinear function, as samples $(\mathbf{x}_k, \mathbf{t}_k = t(\mathbf{x}_k^1, \mathbf{x}_k^2), y_k)$ are available, with a set of kernel functions. This parameter estimation problem can be expressed as a state-space problem (1). Note that here f does not depend on time, but we think that this approach can be easily extended to nonstationary function approximation. In this paper the approximation is of the form

$$\hat{f}_\theta(\mathbf{x}) = \sum_{i=1}^{p} \alpha_i K_{\sigma_i^1}(\mathbf{x}^1, \mathbf{x}_i^1) K_{\sigma_i^2}(\mathbf{x}^2, \mathbf{x}_i^2), \text{ with} \qquad (5)$$

$$\theta = [(\alpha_i)_{i=1}^p, \left((\mathbf{x}_i^1)^T\right)_{i=1}^p, (\sigma_i^1)_{i=1}^p, \left((\mathbf{x}_i^2)^T\right)_{i=1}^p, (\sigma_i^2)_{i=1}^p]^T$$

$$\text{and } K_{\sigma_i^j}(\mathbf{x}^j, \mathbf{x}_i^j) = \exp(-\frac{\|\mathbf{x}^j - \mathbf{x}_i^j\|^2}{2(\sigma_i^j)^2}),\ j = 1, 2$$

Note that $K_{\sigma_i}(\mathbf{x}, \mathbf{x}_i) = K_{(\sigma_i^1, \sigma_i^2)}\left((\mathbf{x}^1, \mathbf{x}^2), (\mathbf{x}_i^1, \mathbf{x}_i^2)\right) = K_{\sigma_i^1}(\mathbf{x}^1, \mathbf{x}_i^1) K_{\sigma_i^2}(\mathbf{x}^2, \mathbf{x}_i^2)$ is a product of kernels, thus it is a kernel. The optimal number of kernels and a good initialisation for the parameters have to be determined first.

To tackle this initial difficulty, the dictionary method is used in a preprocessing step. A prior $\sigma_0 = (\sigma_0^1, \sigma_0^2)$ on the Gaussian width is first put (the same for each kernel). Then a set of N random points is sampled uniformly from $\mathcal{X}$. Finally, these samples are used to compute the dictionary. A set of p points $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_p\}$ is thus obtained such that $\varphi(\mathcal{X}) \simeq \text{Span}\{\varphi_{\sigma_0}(\mathbf{x}_1), \ldots, \varphi_{\sigma_0}(\mathbf{x}_p)\}$ where φ_{σ_0} is the mapping corresponding to K_{σ_0}. Note that even though this sparsification procedure is offline, the proposed algorithm (the regression part) is online. Moreover, no training sample is required for this preprocessing step, but only classical prior which is anyway required for the Bayesian filter (σ_0), one sparsification parameter ν and bounds for $\mathcal{X}$.

A SPKF is then used to estimate the parameters. As for any Bayesian approach an initial prior on the parameter distribution has to be put. The initial parameter vector follows

a Gaussian law, that is $\theta_0 \sim \mathcal{N}(\bar{\theta}_0, \Sigma_{\theta_0})$, where

$$\bar{\theta}_0 = [\alpha_0, \ldots, \alpha_0, \mathcal{D}^1, \sigma_0^1, \ldots, \sigma_0^1, \mathcal{D}^2, \sigma_0^2, \ldots, \sigma_0^2]^T, \text{ and}$$
$$\Sigma_{\theta_0} = \text{diag}(\sigma^2_{\alpha_0}, \ldots, \sigma^2_{\mu_0^1}, \ldots, \sigma^2_{\sigma_0^1}, \ldots, \sigma^2_{\mu_0^2}, \ldots, \sigma^2_{\sigma_0^2}, \ldots)$$

In these equations, α_0 is the prior mean on kernel weights, $\mathcal{D}^j = [(\mathbf{x}_1^j)^T, \ldots, (\mathbf{x}_p^j)^T]$ is derived from the dictionary computed in the preprocessing step, σ_0^j is the prior mean on kernel deviation, and $\sigma^2_{\alpha_0}$, $\sigma^2_{\mu_0^j}$ (which is a row vector with the variance for each component of $\mathbf{x}_i^j$, $\forall i$), $\sigma^2_{\sigma_0}$ are respectively the prior variances on kernel weights, centers and deviations. All these parameters (except the dictionary) have to be set up beforehand. Let $|\theta| = q = (n+m+3)p$. Note that $\bar{\theta}_0 \in \mathbb{R}^q$ and $\Sigma_{\theta_0} \in \mathbb{R}^{q \times q}$. A prior on noises has to be put too: more precisely, $\mathbf{v}_0 \sim \mathcal{N}(0, R_{v_0})$ where $R_{v_0} = \sigma^2_{v_0} I_q$, I_q being the identity matrix, and $n \sim \mathcal{N}(0, R_n)$, where $R_n = \sigma_n^2$. Note that here the observation noise is also structural. In other words, it models the ability of the parameterization $\hat{f}_\theta$ to approximate the true function of interest f. Then, a SPKF update (which updates the sufficient statistics of the parameters) is simply applied at each time step, as a new training sample $(\mathbf{x}_k, \mathbf{t}_k, y_k)$ is available.

A specific form of SPKF is used, the so-called Square-Root Central Difference Kalman Filter (SR-CDKF) parameter estimation form. This implementation uses the fact that for this parameter estimation problem, the evolution equation is linear and the noise is additive. This approach is computationally cheaper: the complexity per iteration is $O(|\theta|^2)$, whereas it is $O(|\theta|^3)$ in the general case. See [9] for details.

A last issue is to choose the artificial process noise. Formally, since the target function is stationary, there is no process noise. But the parameter space has to be explored in order to find a good solution, that is an artificial process noise must be introduced. Choosing this noise is still an open research problem. Following [9] a Robbins-Monro stochastic approximation scheme for estimating innovation is used. That is the process noise covariance is set to

$$R_{v_k} = (1-\alpha) R_{v_{k-1}} + \alpha K_k \left(y_k - g_{\hat{f}_{\bar{\theta}_{k-1}}}(\mathbf{x}_k, \mathbf{t}_k) \right)^2 K_k^T$$

Here α is a forgetting factor set by the user, and K_k is the Kalman gain obtained during the SR-CDKF update. This approach is summarized in Table 2.

4. PRELIMINARY RESULTS

In this section the results of our preliminary experiments are presented. The proposed artificial problem is quite arbitrary, it has been chosen for its interesting nonlinearities so as to emphasize on the potential benefits of the proposed approach. More results focusing on the application of this method to reinforcement learning are presented in [12].

Table 2. Proposed algorithm

inputs: $\nu, N, \alpha_0, \sigma_0^j, \sigma_{\alpha_0}, \sigma_{\sigma_0^j}, \sigma_{\mu_0^j}, \sigma_{v_0}, \sigma_n$
outputs: $\bar{\theta}, \Sigma_\theta$

Compute dictionary;
$\forall i \in \{1 \ldots N\}$, $\mathbf{x}_i \sim \mathcal{U}_\mathcal{X}$
Set $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
$\mathcal{D}$ =Compute-Dictionary(X, ν, σ_0)

Initialization:
Initialize $\bar{\theta}_0$, Σ_{θ_0}, R_{n_0}, R_v

for $k = 1, 2, \ldots$ **do**
  Observe $(\mathbf{x}_k, \mathbf{t}_k, y_k)$
  SR-CDKF update:
  $[\bar{\theta}_k, \Sigma_{\theta_k}, K_k] =$
    SR-CDKF-Update$(\bar{\theta}_{k-1}, \Sigma_{\theta_{k-1}}, \mathbf{x}_k, \mathbf{t}_k, \ldots$
      $\ldots y_k, R_{v_{k-1}}, R_n)$
  Artificial process noise update:
  $R_{v_k} = (1-\alpha) R_{v_{k-1}} + \alpha K_k (y_k - g_{\hat{f}_{\bar{\theta}_{k-1}}}(\mathbf{x}_k, \mathbf{t}_k))^2 K_k^T$
end for

Let $\mathcal{X} = [-10, 10]^2$ and f be a 2d nonlinear cardinal sine:

$$f(x^1, x^2) = \frac{\sin(x^1)}{x^1} + \frac{x^1 x^2}{100}$$

Let the observed nonlinear transformation be

$$t(x^1, x^2) = 10 \tanh(\frac{x^1}{7}) + \sin(x^2)$$

Recall that this analytical expression is only used to generate observations in this experiment but it is not used in the regression algorithm. Let the nonlinear mapping g be

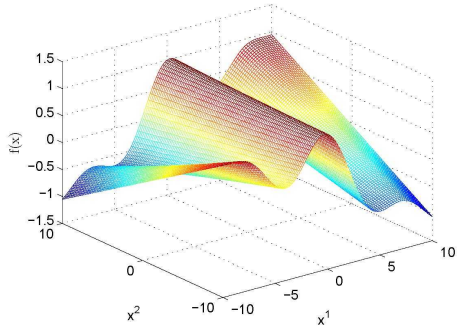
$$g_f\left(x^1, x^2, t(x^1, x^2)\right) = f(x^1, x^2) - \gamma \max_{z \in \mathcal{X}^2} f\left(t(x^1, x^2), z\right)$$

with $\gamma \in [0, 1[$ a predefined constant. This is a specific form of the Bellman equation [3] with the transformation t being a deterministic transition function. Recall that this function is of special interest in optimal control theory. The associated state-space formulation is thus

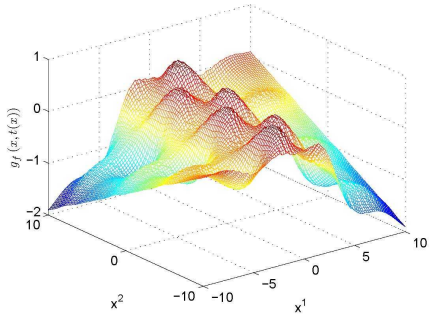
$$\begin{aligned} \theta_{k+1} &= \theta_k + \mathbf{v}_k \\ y_k &= \hat{f}_{\theta_k}(x_k^1, x_k^2) - \gamma \max_{z \in \mathcal{X}^2} \hat{f}_{\theta_k}\left(t(x_k^1, x_k^2), z\right) + n_k \end{aligned} \tag{6}$$

4.1. Problem statement and settings

At each time step k the filter observes $\mathbf{x}_k \sim \mathcal{U}_\mathcal{X}$ ($\mathbf{x}_k$ uniformly sampled from $\mathcal{X}$), the transformation $t_k = t(x_k^1, x_k^2)$ and $y_k = f(\mathbf{x}_k) - \gamma \max_{z \in \mathcal{X}^2} f(t_k, z)$. Once again the regressor does not have to know the function t analytically, it just has to observe it. The function f is shown on Fig. 1(a) and its nonlinear mapping on Fig. 1(b). For these experiments the algorithm parameters were set to $N = 1000$,

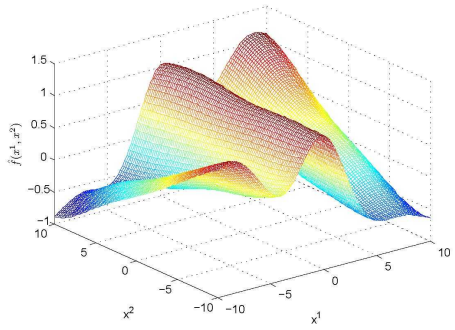


(a) 2-dimensional nonlinear cardinal sine.

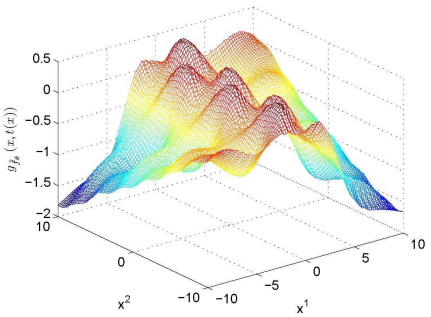


(b) Nonlinear mapping observed by the regressor.

Fig. 1. 2-dimensional nonlinear cardinal sine and its observed nonlinear mapping.



(a) Approximation of $f(x)$.



(b) Nonlinear mapping calculated from $\hat{f}_\theta(x)$

Fig. 2. Approximation of $f(x)$and associated approximated nonlinear mapping.

$\sigma_0^j = 4.4$, $\nu = 0.1$, $\alpha_0 = 0$, $\sigma_n = 0.05$, $\alpha = 0.7$, and all variances ($\sigma^2_{\alpha_0}$, $\sigma^2_{\mu_0^j}$, $\sigma^2_{\sigma_0^j}$, $\sigma^2_{n_0}$) to 0.1, for $j = 1, 2$. The factor γ was set to 0.9. Note that this empirical parameters were not finely tuned.

4.2. Quality of regression

Because of the special form of g_f, any bias added to f still gives the same observations and it may exist other invariances: the root mean square error (RMSE) between f and $\hat{f}$ should not be used to measure the quality of regression. Instead the nonlinear mapping of f is computed from its estimate $\hat{f}$ and is then used to calculate the RMSE. The quality of regression is thus measured with the following RMSE: $(\int_{\mathcal{X}}(g_f(\mathbf{x}, t(\mathbf{x})) - g_{\hat{f}_\theta}(\mathbf{x}, t(\mathbf{x})))^2 d\mathbf{x})^{\frac{1}{2}}$. As it is computed from $\hat{f}$, it really measures the quality of regression, and as it uses the associated nonlinear mapping, it will not take into account the bias. Practically it is computed on 10^4 equally spaced points. The function g_f is what is observed (Fig. 1(b)) and it is used by the SPKF to approximate the function f (Fig. 1(a)) by $\hat{f}_\theta$ (Fig. 2(a)). We compute $g_{\hat{f}_\theta}$ from $\hat{f}_\theta$ (Fig. 2(b)) and use it to compute the RMSE.

As the proposed algorithm is stochastic, the results have been averaged over 100 runs. Fig. 3 shows an errorbar plot, that is mean $\pm$ standard deviation of RMSE. The average number of kernels was 26.55 ± 1.17. This results can be compared with the RMSE directly computed from $\hat{f}$, that is $(\int_{x\in\mathcal{X}}(f(\mathbf{x}) - \hat{f}_\theta(\mathbf{x}))^2 d\mathbf{x})^{\frac{1}{2}}$, which is illustrated on Table 3. There is more variance and bias in these results because of the possible invariances of the considered nonlinear mapping. However one can observe on Fig. 2(a) that a good approximation is obtained.

Table 3. RMSE (mean $\pm$ deviation) of $g_{\hat{f}_\theta}$ and $\hat{f}_\theta$ as a function of number of samples.

	500	1000	1500
$g_{\hat{f}_\theta}$	0.072 ± 0.022	0.031 ± 0.005	0.024 ± 0.003
$\hat{f}_\theta$	0.296 ± 0.149	0.108 ± 0.059	0.066 ± 0.035

Thus the RMSE computed from g_f is a better quality measure. As far as we know, no other method to handle such a problem has been published so far, thus comparisons with other approaches is made difficult. However the order of magnitude for the RMSE obtained from g_f is comparable with the state-of-the-art regression approaches when the function of interest is directly observed. This is demonstrated on Table 4. Here the problem is to regress a linear 2d cardinal sine $f(x^1, x^2) = \frac{\sin(x^1)}{x^1} + \frac{x^2}{10}$. For the proposed algorithm, the nonlinear mapping is the same as considered before. We compare RMSE results (10^3 training points) to

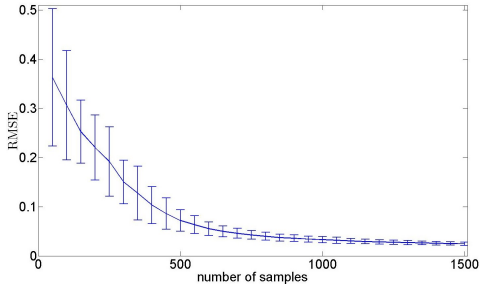


Fig. 3. RMSE ($g_{\hat{f}_\theta}$) averaged over 100 runs.

Kernel Recursive Least Squares (KRLS) and Support Vector Regression (SVR). For the proposed approach, the approximation is computed from nonlinear mapping of observations. For KLRS and SVR, it is computed from direct observations. Notice that SVR is a batch method, that is it needs the whole set of samples in advance. The target of the regressor is indeed g_f, and we obtain the same order of magnitude (however much nonlinearities are introduced for our method and the representation is more sparse). The RMSE on $\hat{f}$ is slightly higher for this contribution, but this can be mostly explained by the invariances (as bias invariance) induced by the nonlinear mapping.

Table 4. Comparative results (KRLS and SVR results are reproduced from [7]). The first column gives RMSE (from $\hat{f}$ and $g_{\hat{f}}$ for the proposed algorithm, from $\hat{f}$ for the two others), and the second one the percentage of support vector/dictionary vectors from the training sample used.

	test error		
Method	$\hat{f}$	$g_{\hat{f}}$	%DV/SVs
Proposed Alg.	2.45×10^{-1}	4.5×10^{-2}	2.6%
KRLS	1.5×10^{-2}		7%
SVR	5.5×10^{-2}		60%

5. CONCLUSIONS AND FUTURE WORKS

An approach allowing to regress a function of interest f from observations which are obtained through a nonlinear mapping of it has been proposed. The regression is online, kernel-based, nonlinear and Bayesian. A preprocessing step allows to achieve sparsity of representation. This method has proven to be effective on an artificial problem and reaches good performance although much more nonlinearities are introduced. This approach is planned to be extended to stochastic transformation function (the observed part of nonlinearities) in order to solve a more general form of the Bellman equation. The adaptive artificial process noise is planned to be extended, and an adaptive observation noise is on study. Finally, the approximation $\hat{f}_\theta$ being a mapping of the random variable θ, it is a random variable. An uncertainty information can thus be derived from it and is planned to be linked to the quality of regression. Yet the uncertainty reduction occurring when acquiring more samples (thus more information) could be quantified.

6. REFERENCES

[1] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press, 2001.

[2] C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, New York, USA, 1995.

[3] R. Bellman, *Dynamic Programming*, Dover Publications, sixth edition, 1957.

[4] V. N. Vapnik, *Statisical Learning Theory*, John Wiley & Sons, Inc., 1998.

[5] C. M. Bishop and M. E. Tipping, "Bayesian Regression and Classification," in *Advances in Learning Theory: Methods, Models and Applications*. 2003, vol. 190, pp. 267–285, OS Press, NATO Science Series III: Computer and Systems Sciences.

[6] J. Vermaak, S. J. Godsill, and A. Doucet, "Sequential Bayesian Kernel Regression," in *Advances in Neural Information Processing Systems 16*. 2003, MIT Press.

[7] Y. Engel, S. Mannor, and R. Meir, "The Kernel Recursive Least Squares Algorithm," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2275–2285, 2004.

[8] E. A. Wan and R. van der Merwe, "The Unscented Kalman Filter for nonlinear estimation," in *Adaptive Systems for Signal Processing, Communications, and Control Symposium*, 2000, pp. 153–158.

[9] R. van der Merwe and E. A. Wan, "Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models," in *Workshop on Advances in Machine Learning*, Montreal, Canada, June 2003.

[10] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, Series D, pp. 35–45, 1960.

[11] Z. Chen, "Bayesian Filtering : From Kalman Filters to Particle Filters, and Beyond," Tech. Rep., Adaptive Systems Lab, McMaster University, 2003.

[12] Matthieu Geist, Olivier Pietquin, and Gabriel Fricout, "Bayesian Reward Filtering," in *European Workshop on Reinforcement Learning (EWRL 2008)*, Lille (France), 2008, Lecture Notes in Artificial Intelligence, Springer Verlag, to appear.